

02-01-00

A

N. D. Griffeth 1-2-10-3

IN THE UNITED STATES  
PATENT AND TRADEMARK OFFICE

## PATENT APPLICATION

INVENTORS: Nancy D. Griffeth      David Lee  
                  Ruibing Hao            Rakesh Kumar Sinha

CASE: 1-2-10-3

TITLE: GENERATION OF TEST SUITES FOR INTEROPERABILITY OF REACTIVE  
COMMUNICATION SYSTEMS

ASSISTANT COMMISSIONER FOR PATENTS  
WASHINGTON, D.C. 20231

SIR: Enclosed are the following papers relating to the above-named application for patent:

Specification, Claims and Abstract - 33 pages  
17 sheets of drawing FIGS. 1-17  
Declaration and Power of Attorney (Declaration of N. D. Griffeth to follow)  
Return receipt postcard


CLAIMS AS FILED				
	NO. FILED	NO. EXTRA	RATE	CALCULATIONS
Total Claims	2 - 20 =	0	x \$18 =	\$ 0
Independent Claims	2 - 3 =	0	x \$78 =	\$ 0
Multiple Dependent Claim(s), if applicable	0		\$260 =	\$ 0
Basic Fee				\$ 690

TOTAL FEE: \$ 690

Please file the application and charge **Lucent Technologies Inc. Deposit Account No. 12-2325** the amount of \$690 to cover the filing fee. Duplicate copies of this letter are enclosed. In the event of non-payment or improper payment of a required fee, the Commissioner is authorized to charge or to credit **Deposit Account No. 12-2325** as required to correct the error.

Please address all correspondence to **Law Office of Leo Zucker, 50 Main Street, White Plains, New York 10606-1975**. Telephone calls should be made to Leo Zucker, Esq. at Area Code (914) 761-7799.

Respectfully,

  
LEO ZUCKER  
Reg. No. 27,608, Attorney for Applicants

White Plains, New York

EXPRESS MAIL LABEL: EK 196 467 483 US

MAIL DATE: January 31, 2000

01/31/00



jc675 U.S. PTO  
09/495036



01/31/00

**GENERATION OF TEST SUITES FOR INTEROPERABILITY OF  
REACTIVE COMMUNICATION SYSTEMS**

**BACKGROUND OF THE INVENTION**

5      Field of the Invention

          This invention relates to a method of generating a set of test sequences to confirm proper operation of interconnected communication systems.

Discussion of the Known Art

10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201

Such networks are expected to handle not only voice, but video and other information and data as well. It is therefore important that existing data-centric and public switched telephone networks (PSTNs) be capable of operation when connected with one another through so-called "gateways". The ability of such networks or systems to operate properly when inter-connected is typically referred to as system "interoperability".

With respect to providing voice calls over an IP network ("Voice over IP" or "VoIP"), the first VoIP systems provided voice communications over a local area network (LAN) for end-users seated at personal computers. This proved unsatisfactory for several reasons. For example, voice quality was poor due to a "best-effort" standard for data packet delivery in the LAN environment. Also, there was no way to direct a voice call to a user connected to a PSTN, thus limiting the number of possible calls. And, a called end-user PC might not be connected to the network when the call was placed.

Because some providers believed IP telephony might circumvent long distance user rates due to the absence of tariffs on local Internet access, local user access gateways to PSTNs were created in each locality to be served, with a private Internet providing a long-distance backbone for voice calls. LEC's became interested in a similar strategy, but instead of

using a gateway to connect local lines to an IP network, gateways were used to connect their central offices to an IP network, thus replacing a circuit-switched network backbone with an IP network. Finally, corporations maintaining large data networks became  
5 interested in combining their voice and data networks by running Voice over IP.

In view of the above, the existing diversity of VoIP implementations results in diverse communications systems that may not be able to interoperate when connected together. In  
10 order for these existing communication systems to remain effective, they must be able to interwork with the PSTN and with other VoIP products.

### The Standards

Three important Internet telephone standards now exist.  
15 Initially, Standard H.323 for Internet Telephony reflected the LAN bias of earlier Internet telephony systems. Subsequent standards address use of packet voice over wide-area networks and interwork with the PSTN.

Standard H.323 version 2 is a current dominant standard  
20 for commercial products. A second standard, Session Initiation Protocol (SIP), is a proposed standard of the Internet

Engineering Task Force. A third standard, Media Gateway Control Protocol (MGCP), has also been proposed to provide interworking between Internet telephone and the PSTN.

5 The H.323 Standard is both an architecture and a standard for Voice over IP. It is an umbrella standard, specifying a collection of standards to be used by the components of the architecture which divides required functionality among gatekeepers, gateways, multipoint control units (MCUs), and endpoints.

10 Specifically, endpoints are the initial creators and ultimate recipients of information streams. They need not comply with any part of the H.323 protocol, but if they do not, they must communicate with the network through a gateway that does comply with the protocol. An endpoint that complies with H.323  
15 is called an H.323 terminal.

An H.323 gateway provides communication between an H.323 network and a non-H.323 device or network. It may also provide communication with a second H.323 network.

20 An H.323 gatekeeper handles address translations, for example, between telephone numbers and IP addresses, and controls access to the network. A gatekeeper controls admission to a

zone. Gateways, MCU's, and H.323 terminals must register with a gatekeeper, if one is present in their zone, and obtains permission to join a call. Optionally, call signaling may also go through the gatekeeper. There is no gatekeeper-to-gatekeeper standard, so that communication between multiple zones must be managed in an ad hoc fashion.

A multipoint control unit provides conferencing capabilities in an H.323 network. It includes a multipoint controller, and (optionally), one or more multipoint processors. The multipoint controller handles signaling that determines who participates in a conference, and what information streams they send and receive. The multipoint processors provide centralized processing of information streams (audio, video, or data) from various parties to a conference. The processing can be mixing, switching, or anything supporting a specific conference connection.

An H.323 terminal or gateway uses a Registration, Admission, and Status (RAS) channel defined in the standard to register with a gatekeeper, to locate other gateways and terminals, and to request permission to start or join a call. It may also route the call signaling through the gatekeeper. The call signaling protocol is a Q.931 subset, also defined in Standard H.225.0.

The actual media connections in a call and their types, are negotiated directly between gateways and/or endpoints using Standard H.245. Once the endpoints agree, transport connections are also set up using the H.245 standard. Transport connections  
5 can use any of a number of standards. For audio, Standard G.711 for uncompressed voice is required, but not always implemented. Other, optional audio Standards are G.723.1 and G.729. For video, Standard H.261 is required; and Standard H.263 is optional. For data, Standard T.120 is required.

10 Because of the many choices available when building an H.323-compliant system, and since requirements for communication between zones are not specified, it is quite likely that H.323-compliant systems manufactured by different vendors will not interoperate. Further, the number of configurations to be  
15 tested to determine system interoperability, becomes relatively large.

The Session Initiation Protocol (SIP) identifies the following functionality which is needed to set up multimedia conferences:

20 User Location: Determination of the end system to be used for communication. A location server provides the user location information, in response to a request from a SIP server

or proxy. Users may also register their locations with a SIP server, using a REGISTER method.

User Availability: Determination of the willingness of a called party to engage in communications. An end-user wishing to set up a call with another end-user, or to add it to a conference, invokes an INVITE method on the user's SIP server. Possible responses include "Success" and "Redirection".

User Capabilities: Determination of the media and media parameters to be used. To determine what media and media parameters will be used for a call, the calling system invokes an OPTION method on the SIP server of the called party. This is an inquiry as to what are the capabilities of the called party.

Call Setup: "Ringing", establishment of call parameters at both called and calling party.

Inoperability of SIP implementations is less problematic than that of H.323 implementations. Interoperation of SIP user agents and servers with H.323 networks remains problematic, however. A type 7R/E Programmable Feature Server, available from Lucent Technologies Inc., provides protocol interworking through device servers that interface to specific protocols.



The Media Gateway Control Protocol (MGCP) is a combination of two earlier standards, IP Device Control (IPDC) and SGCP (Simple Gateway Control Protocol SGCP). The protocol provides for media gateways that convert signals from TDM to packet, and back, and for call agents that control the behavior of media gateways. This is a master/slave arrangement, with the call agent as the master and the gateway as the slave. Synchronization between various call agents involved in a call must be handled by other means.

MGCP is thus used as an internal protocol within a distributed system that appears as a single VoIP gateway to an external H.323 network. A call agent may signal to other call agents using the RAS and H.225.0 protocols. This provides means to interwork H.323 networks with IP networks.

Accordingly, there is currently a need to verify that when two communication systems are connected for interworking, such as to provide, e.g., VoIP service, the connected system behavior will be as expected from interactions with one or more components of the integrated system under test. Specifically, there is a need for a method in the form of, e.g., portable software, that will generate test sequences which, if successfully implemented, will assure both branch and full path coverages for the connected systems.

## SUMMARY OF THE INVENTION

According to the invention, a method of generating a set of test sequences to determine interoperability of communication systems when connected to one another, includes defining gateways between the connected systems and corresponding end users of the connected systems. Certain transitions are determined that involve operation of more than one gateway associated with a calling user and a called user, and the connected systems are tested for operation at those transitions.

According to another aspect of the invention, a method of generating a set of test sequences to determine interoperability of communication systems when connected to one another, includes defining a state machine corresponding to the connected systems, the machine having a number of defined states, and transitions between the states. Transitions that involve operation of more than one system gateway associated with a calling user and a called user are determined, and the connected systems are tested for operation at at least some of the transitions.

For a better understanding of the invention, reference is made to the following description taken in conjunction with the accompanying drawing and the appended claims.

## BRIEF DESCRIPTION OF THE DRAWING

In the drawing:

FIG. 1 is a block diagram of a model showing voice end-users connected to one another by an IP Network through associated gateways;

FIG 2. is a typical finite state machine (FSM) representation of the interconnected model in FIG. 1;

FIGS. 3-9 show various states, and transitions between the states in the FSM of FIG. 2;

FIGS. 10-13 together represent a first step toward determining which transitions are to be tested for interoperability of interconnected communication systems;

FIG. 14 represents a second step toward determining a set of tests for system interoperability;

FIG. 15 represents a third step toward determining a set of tests for system interoperability;

FIG. 16 is a block diagram representing criteria for a

set of tests to ensure adequate coverage for interconnected communication systems; and

FIG. 17 is a diagram representing a software program that produces test sequences for interconnected communication systems.

5

## DETAILED DESCRIPTION OF THE INVENTION

Interoperability testing is concerned only with failures that may occur when two different systems are connected or coupled with one another to interoperate. It is assumed that two systems operate correctly when running independently, or when coupled to other identical systems. Potential interoperation failures to be checked are those due to interoperation between two independently designed or developed systems.

A model 10 for a general configuration of interoperating communication systems, such as Voice over Ip systems, is shown in FIG. 1. The figure shows two end-users 12, 14 that want to communicate. Each user can access the systems through a corresponding gateway 16, 18. The two gateways 16, 18 communicate with each other using a defined protocol to decide whether and how to start the communication between the end-users 12, 14. The protocol takes the gateways 16, 18 through various states as they negotiate concerning the desired communication. The end users 12, 14 may wish to communicate by voice, or to exchange other information in either analog or digital form.

It is important to note that interoperability errors, if any, will be introduced only when the gateways 16, 18 actually communicate with one another about a call. Local activities involved in the protocol, e.g., obtaining information from either

end-user 12 or 14 can be ignored. Interactions or "transitions" that can be ignored for interoperability purposes are referred to as "white", and all other transitions are referred to as "black". A white transition is purely local, that is, it reads the state of only one gateway 16 or 18, and writes the same state. A black transition involves both gateways 16 and 18 because the transition reads the states of both of the gateways, or reads the state of one and writes the state of the other.

For example, an "off-hook" transition from an idle state of a user's telephone to a dialing state of the same user's telephone is white, because the transition involves only an originating gateway and its associated end user. A "dial" transition from "dialing" of a calling user's telephone to "ringing" of a called user's telephone is black, because the states change in both gateways 16 and 18. Similarly, an off-hook transition that connects the call changes states in both gateways 16, 18 from ringing to connected. Accordingly, test cases that correspond only to sequences involving black edges are generated, i.e., those sequences that occur while a call is under negotiation between the two gateways 16 and 18.

Expected system behavior is modeled in accordance with a finite state machine (FSM) 50, typical vertices (nodes) and edges of which are shown in FIG. 2. The FSM 50 of FIG. 2 has 21 states (nodes) and a total of 68 transitions between the states,

as defined in FIGS. 3-9. A transition from a first state to a second state is identified by locating the two ordered states on the first line of one of the 68 transitions in FIGS. 3-9.

Ideally, all possible execution sequences or "scenarios" should be covered. Because the transition diagram of the FSM 50 is a directed graph, covering all possible execution sequences requires that all branches and all possible paths be tested. Criteria for ruling out "redundant" scenarios are given further below, however.

Generally, the transition diagram will contain cycles, and, therefore, will have an infinite number of distinct paths. Therefore, the test generation process includes the following three steps:

#### Test Generation Process

Step 1: Generate all possible acyclic paths, i.e., paths without repeated vertices. See FIGS. 10 to 13.

Step 2: Generate all possible simple cycles, i.e., cycles that do not contain any smaller cycles. See FIG. 14; and

Step 3: "Combine" the paths (from Step 1) and cycles (from Step 2) to generate a final set of paths. See FIG. 15.

The number of paths and cycles generated in the first two steps is finite. Various criteria for removing redundant acyclic paths and redundant simple cycles are given later below.

With respect to Steps 1 and 2, all strongly connected components (SCC) of the transition graph are first found. This has two advantages; (a) we know that any cycle is completely contained inside a SCC, so, Step 2 can be performed by looking at each SCC in turn and finding all simple cycles within the SCC, and (b) we can "shrink" each SCC into a node and obtain a Directed Acyclic Graph (DAG), i.e., a graph without any cycles. This translates into a two-phase process for Step 1. First, generate all acyclic paths on the resulting DAG, and then replace each SCC on any given path with a set of acyclic paths within the SCC.

The goal of the "combine" process in Step 3 is to generate a finite number of paths that cover all scenarios of interest. This is done first by including all the acyclic paths. Then additional cyclic paths are generated as follows.

For each acyclic path  $P$ , find all the cycles that share a node with  $P$ . Let these cycles be  $C_1, C_2, \dots, C_k$ , and let  $v_i$ ,  $1 \leq i \leq k$ , be a node common to  $C_i$  and  $P$ . Then generate a new (cyclic) path by replacing node  $v_i$  in  $P$  with cycle  $C_i$ .



## Next-Transition-Tree

A simple data structure, next-transition-tree, which is convenient for Steps 1 and 2, is now described. See FIG. 12.

The data structure can be defined for any graph but, in the present application, the graph is always a SCC.

For any node  $v$ , next-transition-tree( $v$ ) stores all acyclic paths from  $v$  to other vertices in its SCC. The tree has  $v$  as its root, and its height is equal to the number of nodes in the SCC containing  $v$ . Children of  $v$  are all the nodes in its SCC that have a direct edge from  $v$ . In general, the children of any node  $u$  are all the nodes in its SCC that have a direct edge from node  $u$ . Note that a node may appear multiple times in this tree. That is, each node has a label where the labels are not necessarily unique. The root node has label  $v$ . A node labeled  $u$  has as many children as the outdegree of  $u$  (number of edges leaving  $u$ ) in its SCC, and these children are labeled with the corresponding nodes in the SCC.

For ease of illustration, assume that a separate next-transition-tree is maintained for each node in the graph. The actual implementation may have many shared pieces among next-transition-trees belonging to the same SCC.

## Generating All Simple Cycles

All simple cycles containing any node  $v$  are generated as follows: Consider all paths in  $\text{next-transition-tree}(v)$  that contain another (than root) instance of node  $v$ . In any such path, the path segment from the root to the first (closed from root) occurrence of  $v$  corresponds to a cycle containing  $v$ . This path segment will correspond to a "simple" cycle if it does not contain any repeated nodes. Therefore, the process includes finding all such path segments, and all simple cycles containing  $v$  are generated. As mentioned, this is a simplified description of an actual implementation, which should not maintain any paths with repeated nodes in the next-transition-tree.

One way to generate all simple cycles in a SCC is to consider its vertices in some order:  $v_1, v_2, \dots, v_k$ . Generate all simple cycles containing the node  $v_1$ . Then generate all simple cycles containing the node  $v_2$  that don't contain node  $v_1$ . This can be accomplished by modifying the above procedure so that path segments containing the node  $v_1$  are ignored. All simple cycles containing the node  $v_3$  that don't contain nodes  $v_1$  or  $v_2$ , are then generated, and so on.

## Generating All Acyclic Paths

As mentioned earlier, all the strongly connected components (SCC) of the transition graph must be found. A "component graph" where each node represents a SCC, and an edge from node u to node v represents an edge (in the original transition graph) from a node belonging to u's SCC to a node belonging to v's SCC, is then constructed. This component graph is a Directed Acyclic Graph (DAG), i.e., a graph without any cycles. See A.V. Aho, et al., The Design and Analysis of Computer Algorithms (Addison-Wesley 1974), all relevant portions of which are incorporated by reference. All acyclic paths in the DAG are first generated, and then each SCC on any given path is replaced with a set of acyclic paths within the SCC.

### Redundancy

There are a finite number of distinct paths representing potential tests in a DAG. Not all of them need to be tested, however. Rather, a minimal number of tests, which collectively satisfy a coverage criterion for the connected systems and which do not contain any redundant tests, are desirably generated.

The tests must start from a "Start" state, so only

those paths beginning from a source node which has no incoming transitions are considered first. The following redundancy criteria may then be applied.

(R1) Proper prefixes are redundant and may be ignored.

5           Redundancy criterion R1 implies that only those paths terminating at a sink node (a node without an outgoing transition) be considered. Any path not terminating at a sink node is a proper prefix of a path that has been extended to a sink node. Accordingly, only those paths that start from the source node and terminate at one of the sink nodes need to be generated.

The following procedure generates all source-to-sink paths in a DAG.

#### PATHS-IN-DAG

**Input.** a DAG  $G$  with one sources and multiple sinks.

**Output.** all source-sink paths in  $G$ .

```
1  topologically sort nodes in  $G$ :  $s = v_0, v_1, \dots, v_{n-1}$ ;  
2  for  $i = n - 1, \dots, 0$   
3      if  $v_i$  is a sink node then  
4           $p(v_i) = \{\Lambda\}$ ; /* a singleton set of empty path */  
5      else  
6          let outgoing edges from  $v_i$  be:  $w_1, \dots, w_r$ ;  
7           $p(v_i) = \cup_{j=1}^r (v_i, w_j)p(w_j)$ ;  
8  return  $p(v_0)$ 
```

5 The above procedure is bottom-up, starting from one of  
the sink nodes  $t = v_{n-1}$ . When processing a node  $v_i$ , we examine  
all of its outgoing edges  $(v_i, w_j)$  where the paths from  $w_j$  to sink  
nodes have been computed. At Line 7 above, we concatenate edge  
( $v_i, w_j$ ) to each path computed at  $w_j$  and collect it at node  $v_i$ .  
After processing the source node  $s = v_0$ , all of the paths which  
are irredundant and have complete coverage are obtained.

10 A topological sort takes time proportional to the  
number of edges. We charge to the examination and concatenation  
of each edge, which is processed only once at Line 7, and the  
total cost is proportional to the total paths lengths.

15 Proposition 1. The procedure PATHS-IN-DAG constructs  
all source-sink paths. Its time and space requirement is linear  
in output, i.e., the total lengths of all the constructed tests.

Proposition 2. For machines with a reset to source  $s$ ,  
the constructed test sequences are a checking sequence.

#### Generating All Acyclic Paths within a SCC

20 The generation of paths (i.e., tests) for DAGs where  
nodes can be SCC's has been discussed. For each edge connecting  
two SCC's on a path, the edge is replaced with an edge in the

original graph. In general, it will be possible to replace an edge between SCC's with one of several possible edges in the original graph, and a separate path for each choice of replacement edge can be obtained.

5           Each SCC node is also replaced with a set of acyclic paths within the SCC. Assume the incoming and the outgoing edges to the SCC node are on nodes  $u$  and  $v$  of the SCC. The SCC node needs to be replaced with all possible acyclic paths from  $u$  to  $v$  in the SCC. These paths can be generated from next-transition-tree( $u$ ).

10           Replacing any SCC with all possible acyclic path provides exhaustive coverage, but may generate a relatively large number of tests. In certain applications, it may not be necessary to cover the SCC's as thoroughly. Two options used in practice are as follows:

15           Chinese Postman Tour. For an SCC, each edge (transition) is tested at least once and the test sequence length is minimized. Such a path is called Chinese Postman Tour. See A. V. Aho, et al., An Optimization Technique for Protocol  
20 Conformance Test Generation Based on UIO Sequences and Rural Chinese Postman Tours, 39 IEEE Trans. on Communication, No. 11 (Nov. 1991) at 1604-15, which is incorporated by reference.

Checking Sequence. A more thorough coverage is provided by a checking sequence that guarantees the structural isomorphism of the implementation and the specification machines. The length of such a test sequence will be longer than that of a Chinese Postman Tour.

The following procedure summarizes the steps in generating all possible tests:

#### TEST-GENERATION

**Input.** An EFSM with a designated source state

**Output.** Minimal set of tests providing full coverage

- 1 Construct an equivalent FSM and its transition diagram  $G$
- 2 Find all SCC's of  $G$ .
- 3 For each SCC
  - 4       for each node in the SCC
    - 5               construct next-generation-tree
- 6 Shrink each SCC of  $G$  to obtain a DAG  $G'$
- 7 Apply procedure PATHS-IN-DAG on  $G'$  to obtain all acyclic source-sink paths in  $G'$
- 8 Replace each edge and node to obtain the set  $P$  of acyclic paths
- 9 Obtain the set  $C$  of all simple cycles
- 10 Combine the sets  $P$  and  $C$  to obtain the final set of tests

## Additional Redundancy Criteria

The TEST-GENERATION procedure, above, provides exhaustive coverage. The number of tests generated may be enormous even for moderate sized systems, however.

5           As stated earlier, system interoperability errors may occur only when the system gateways actually "talk" to one another concerning a given call. Transitions are therefore labeled as either "white" to connote local activity, or as "black" if they involve both gateways. Thus, for interoperability test generation, white transitions need not be covered, and the following additional redundancy criteria become appropriate.

(R2) Remove all-white test sequences.

(R3) Let  $(u,v)$  be the last black edge in the test sequence.

15 Then replace the path from the source node to node  $u$  with the shortest path between the source node and node  $u$ .

(R4) Let  $(u,v)$  be the first black edge in the test sequence.

Then replace the path from node  $v$  to the sink node with the shortest path between node  $v$  and the sink node.



(R5) If there is a sequence of white edges in which each party separately reaches an idle state, terminate the sequence at the last black edge and then use a shortest path to the state "IdleA, IdleB" at the end of the sequence.

5

Criteria R2, R3 and R4, above, reflect that white transitions are not relevant for interoperability testing. Their only use is to connect relevant black transitions. For example, R3 states that the sequence of transitions before the first black transition is not relevant, so that part may be replaced with the shortest all-white sequence. Criterion R4 is a dual of R3. The last criterion R5 reflects that if both parties reach an "idle" state through a sequence of non-relevant transitions, then nothing concerning system interoperability will happen in the rest of the sequence.

10

15

The following procedure generates all acyclic paths satisfying these criteria:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100

**Input.** transition graph  $G$  of a FSM.

**Output.** complete set of paths according to redundancy criteria  $R1-R5$ .

- ```

1  for each node  $v$  that has an outgoing black edge
2      if there is an all-white path from the source node to  $v$  then
3          add a super-edge from the source node to  $v$ ;
4  delete all outgoing edges (not super-edges) from the source node;
5  for each node  $v$  that has an incoming black edge
6      if there is an all-white path from  $v$  to a sink node then
7          add a super-edge from  $v$  to this sink node;
8  delete all incoming edges (not super-edges) to the sink node;
9  generate all acyclic paths in the resulting graph by procedure PATHS-IN-DAG
   with the modification that no super-edge should follow or precede a white edge;
10 replace each super-edge with the shortest all-white path in the original graph;
11 process each path according to redundancy criterion  $R5$ ;

```

## Generating Test Sequences according to R1-R5

10 ensures that a black transition follows a super-edge. The marker 'super-edge' is replaced on line 10 by the shortest all-white path. This handles redundancy criterion R3. Lines 5-8 perform an analogous function for sink nodes and the last black transitions in the sequence, reflecting redundancy criterion R4.

15 Criterion R2 is satisfied because line 9 ensures there is at least one black transition. Instead of generating all paths and

then processing them according to criterion R5 per line 11, an actual implementation may ensure that the incremental construction of all paths in procedure PATHS-IN-DAG is such that a path in violation of R5 is never generated.

- 5 Proposition 3. The procedure COMPLETE-COVERAGE constructs all acyclic paths according to criteria R1-R5.

#### Generating a Smaller Set of Test Sequences

On most practical systems, we expect the above procedure COMPLETE-COVERAGE to generate a considerably smaller set of test sequences than the procedure TEST-GENERATION. But even such a smaller set may be too large for manual test execution, however. Accordingly, the following describes a set of more restrictive criteria for test generation, concentrating only on black transitions.

- 15 (R6) Generate acyclic paths having only black edges, except that the prefix from the source node, and the suffix to the sink node, are allowed to contain white edges.

(R7) Generate simple cycles having only black edges.

The following procedure ADEQUATE-COVERAGE generates a test-set according to R6 and R7.

*Input.* transition graph  $G$  of a FSM.

*Output.* test generation according to R6 and R7.

1 full coverage.

- 2 for each node  $v$  that has an outgoing black edge
- 3     if there is a path from the source node to  $v$  then
- 4         add a super-edge from the source node to  $v$ ;
- 5 for each node  $v$  that has an incoming black edge
- 6     if there is a path from  $v$  to a sink node then
- 7         add a super-edge from  $v$  to this sink node;
- 8 delete all white edges from the graph;
- /\* the only remaining edges are super-edges or black edges \*/
- 9 generate all acyclic paths in the resulting graph;
- 10 replace each super-edge with the shortest path in the original graph;

The procedure ADEQUATE COVERAGE starts with a graph  $G$ , and transforms it into a graph  $G'$  such that the set of source-sink paths in  $G'$  is the same as the set of source-sink paths in  $G$  with criterion R6. The intuition of the procedure is to delete all white edges except those needed to reach black transitions from the source node, or from the black transitions to the sink nodes. Line 4 maintains a marker 'super-edge' for each source node to black transition path. On line 10, this marker is replaced with the shortest path. Lines 4-6 perform an analogous function for sink nodes. Criteria for adequate coverage are represented in FIG. 16.

Proposition 4. The procedure ADEQUATE-COVERAGE constructs all acyclic paths according to criterion R6.

If automated test execution is available, then the COMPLETE-COVERAGE procedure is more desirable. If tests must be executed by hand, however, then the ADEQUATE-COVERAGE procedure will likely produce a manageable set of test sequences.

5 Alternatively, one may always start with the ADEQUATE-COVERAGE procedure since it does apply to the most critical interoperability behavior. If the systems pass those tests generated by ADEQUATE-COVERAGE, one may obtain the broader coverage provided by the COMPLETE-COVERAGE procedure.

#### EXAMPLE

10 A portable software tool referred to as Interoperability Testing Intelligent System (IT-IS) for automated interoperability test generation, was written in ANSI C and Tcl/Tk. The program includes a graphical user interface (GUI) for user input, and for displaying generated test sequences. The workflow of IT-IS is shown in FIG. 17.

15 The input to IT-IS is an extended FSM (EFSM) description of the composed system behavior. IT-IS first performs reachableness analysis to convert the EFSM into a FSM, e.g., FSM 20 50 in FIG. 2, and then uses different procedures on the FSM to generate the test sequences.

IT-IS was used to generate interoperability test cases for end-user VoIP testing. As shown in FIGS. 3-9, the FSM 50 has 21 states and 68 transitions.

Among the transitions, 24 are colored black, others are white. The shrunk DAG generated by IT-IS from the FSM 50 contains 3 SCC nodes, and only one of them has more than one state. The number of test sequences generated by applying the procedures described above, is shown in the following table.

| Procedure         | Loop-Free Paths | Loops | Final Tests |
|-------------------|-----------------|-------|-------------|
| TEST-GENERATION   | 950             | 424   | 1752        |
| COMPLETE-COVERAGE | 508             | 424   | 908         |
| ADEQUATE-COVERAGE | 16              | 4     | 22          |

The final tests for either of the complete-coverage or the adequate-coverage procedures involve only black transitions.

Interoperability testing is indispensable for integration of reactive communication systems. The presently disclosed procedures are distinguishable over conventional conformance testing techniques in that the procedures focus on the system interfaces, and are not directed solely to individual system implementations or specifications.

While the foregoing description represents preferred embodiments of the invention, it will be obvious to those skilled in the art that various changes and modifications may be made, without departing from the spirit and scope of the invention pointed out by the following claims.

WE CLAIM:

1           1. A method of generating a set of test sequences to  
2 determine interoperability of communication systems when  
3 connected to one another, comprising:

4           defining gateways between the connected systems and  
5 corresponding end users of the connected systems;

6           determining certain transitions that involve operation  
7 of more than one gateway associated with a calling user and a  
8 called user; and

9           testing the connected systems for interoperation at  
10 said certain transitions.

11           2. A method of generating a set of test sequences to  
12 determine interoperability of communication systems when

13 connected to one another, comprising:

14           defining a state machine corresponding to the connected  
15 systems, the machine having a number of defined states, and  
16 transitions between the states;

17           determining which of the transitions involve operations  
18 of more than one system gateway associated with a calling user  
19 and a called user; and



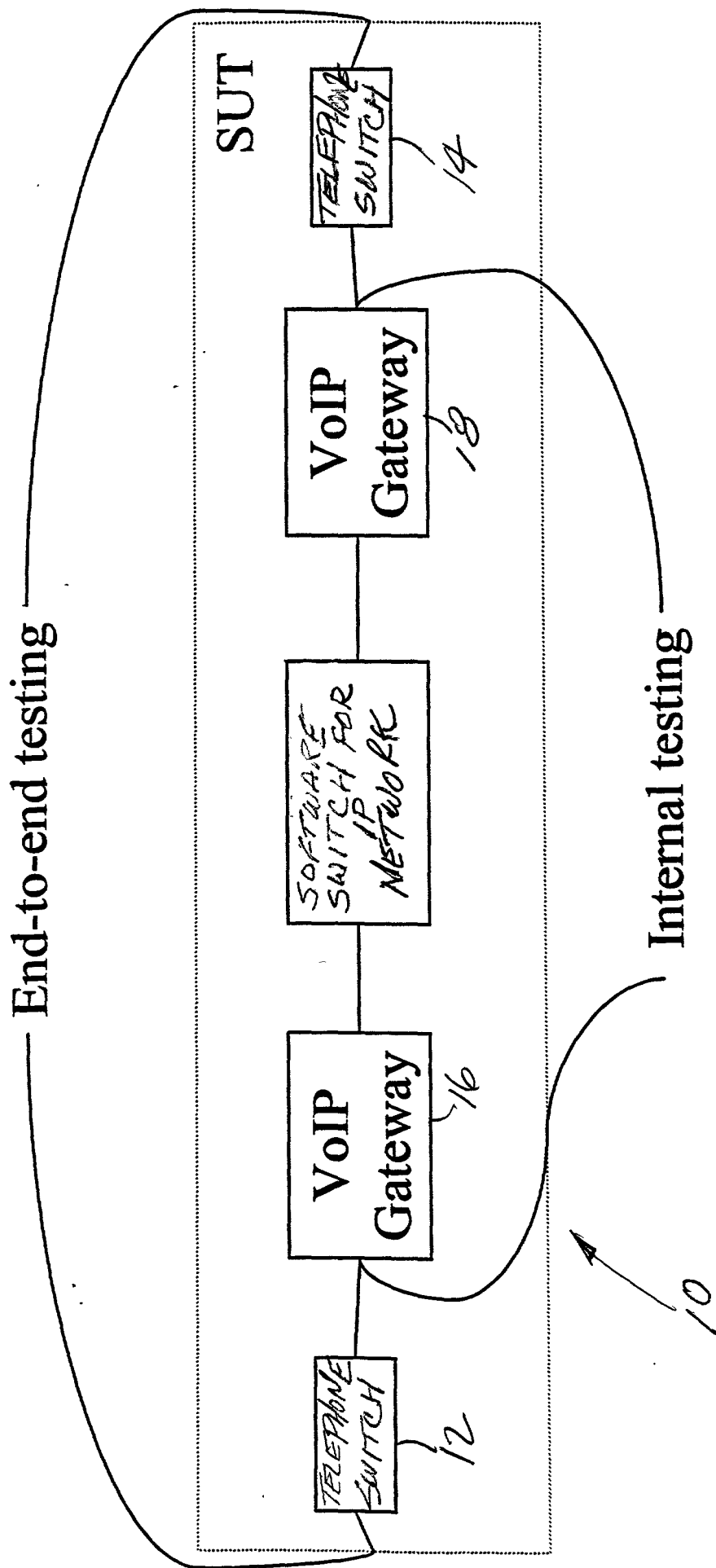
10 testing the connected systems for operation at at least  
11 some of said certain transitions.

GENERATION OF TEST SUITES FOR INTEROPERABILITY OF  
REACTIVE COMMUNICATION SYSTEMS

ABSTRACT OF THE DISCLOSURE

5           A method of generating test sequences for determining  
interoperability of communication systems when connected to one  
another, includes defining gateways between the connected systems  
and corresponding end users of the systems. Certain transitions  
involving operation of more than one gateway associated with a  
10 calling user and a called user, are determined. The connected  
systems are tested for operation at those transitions. In an  
illustrated embodiment, a state machine corresponding to the  
connected systems is defined, the machine having a number of  
defined states, and transitions between the states. Those  
15 transitions involving operation of more than one system gateway  
associated with a calling user and a called user, are determined.  
The connected systems are tested for operation at at least some  
of those transitions.

End-to-end testing



Internal testing

FIG. 1



```

gui fsm
states { 21
  CallingACalledB
  DialingADialingB
  IdleABusyB
  CalledACallingB
  BusyABusyB
  DialingALimboB
  BusyADialingB
  BusyAIdleB
  BusyALimboB
  LimboADialingB
  ConnectedCalledACConnectedCallerB
  LimboABusyB
  IdleADialingB
  LimboAIdleB
  DialingABusyB
  HoldCalledAHoldCallerB
  HoldCallerAHoldCalledB
  DialingAIdleB
  IdleALimboB
  ConnectedCallerACConnectedCalledB
  IdleAIdleB
}

boolean term=false;
boolean IdleA=true;
boolean IdleB=true;
boolean DialingA=false;
boolean DialingB=false;

transitions { 68
  BusyADialingB BusyAIdleB {
    i { {On-hook B} }
    o { {} }
    p { true }
    a { { DialingB=false; IdleB=true } }
  }
  IdleADialingB DialingADialingB {
    i { {Off-hook A} }
    o { {DialTone A} }
    p { true }
    a { {IdleA=false; DialingA=true} }
  }
  BusyADialingB IdleADialingB {
    i { {On-hook A} }
    o { {} }
    p { true }
    a { { IdleA=true } }
  }
  IdleABusyB IdleAIdleB {
    i { {On-hook B} }
    o { {} }
    p { true }
    a { {term=true; IdleB=true} }
  }
}

```

*i = input*  
*o = output*  
*P = predicate*  
*a = action*

Fig. 3

```

5. HoldCallerAHoldCalledB IdleAIdleB {
    i { {On-hook A} }
    o { {} }
    p { true }
    a { {term=true;IdleA=true;IdleB=true} }
}

6. IdleALimboB IdleADialingB {
    i { {Timeout B} }
    o { {DialTone B} }
    p { true }
    a { { DialingB=true } }
}

7. BusyADialingB BusyABusyB {
    i { {Dial B A} }
    o { {LineBusyTone B} }
    p { true }
    a { { DialingB=false } }
}

8. HoldCalledAHoldCallerB IdleAIdleB {
    i { {On-hook B} }
    o { {} }
    p { true }
    a { {term=true;IdleB=true;IdleA=true} }
}

9. DialingAIdleB DialingADialingB {
    i { {Off-hook B} }
    o { {DialTone B} }
    p { true }
    a { { IdleB=false; DialingB=true } }
}

10. IdleADialingB IdleAIdleB {
    i { {On-hook B} }
    o { {} }
    p { true }
    a { {term=true;IdleB=true;DialingB=false} }
}

11. BusyABusyB BusyAIdleB {
    i { {On-hook B} }
    o { {} }
    p { true }
    a { { IdleB=true } }
}

12. DialingADialingB BusyADialingB {
    i { {Dial A B} }
    o { {LineBusyTone A} }
    p { true }
    a { { DialingA=false } }
}

13. IdleAIdleB IdleADialingB {
    i { {Off-hook B} }
    o { {DialTone B} }
    p { !term }
    a { {IdleB=false; DialingB=true} }
}

14. HoldCalledAHoldCallerB IdleADialingB {
    i { {Timeout AB} }
    o { {DialTone B} }
    p { true }
    a { { IdleA=true; DialingB=true } }
}

15. IdleABusyB DialingABusyB {
    i { {Off-hook A} }
    o { {DialTone A} }
    p { true }
    a { { IdleA=false; DialingA=true } }
}

16. CallingACalledB ConnectedCallerAConnectedCalledB {
    i { {Off-hook B} }
    o { {} }
    p { true }
    a { {} }
}

```

P/K. 4

```

17. DialingADialingB DialingABusyB {
    i { { Dial B A } }
    o { { LineBusyTone B } }
    p { true }
    a { { DialingB=false } }
}

18. CalledACallingB IdleALimboB {
    i { { Timeout AB } }
    o { { } }
    p { true }
    a { { IdleA=true } }
}

19. ConnectedCallerAConnectedCalledB IdleALimboB {
    i { { On-hook A } }
    o { { } }
    p { true }
    a { { IdleA=true } }
}

20. ConnectedCallerAConnectedCalledB HoldCallerAHoldCalledB {
    i { { On-hook B } }
    o { { } }
    p { true }
    a { { } }
}

21. IdleAIdleB DialingAIdleB {
    i { { Off-hook A } }
    o { { DialTone A } }
    p { !term }
    a { { IdleA=false; DialingA=true } }
}

22. DialingABusyB BusyABusyB {
    i { { Dial A B } }
    o { { LineBusyTone A } }
    p { true }
    a { { DialingA=false } }
}

23. IdleALimboB IdleAIdleB {
    i { { On-hook B } }
    o { { } }
    p { true }
    a { { term=true; IdleB=true } }
}

24. LimboAIdleB IdleAIdleB {
    i { { On-hook A } }
    o { { } }
    p { true }
    a { { term=true; IdleA=true } }
}

25. HoldCalledAHoldCallerB ConnectedCalledAConnectedCallerB {
    i { { Off-hook A } }
    o { { } }
    p { true }
    a { { } }
}

```

F/16.5

```

}
26. DialingABusyB IdleABusyB {
    i { {On-hook A} }
    o { { } }
    p { true }
    a { { DialingA=false; IdleA=true } }
}

27. DialingABusyB DialingAIdleB {
    i { {On-hook B} }
    o { { } }
    p { true }
    a { { IdleB=true } }
}

28. HoldCallerAHoldCalledB DialingAIdleB {
    i { {Timeout AB} }
    o { {DialTone A} }
    p { true }
    a { { DialingA=true; IdleB=true } }
}

29. DialingAIdleB CallingACalledB {
    i { {Dial A B} }
    o { {AudibleRinging A,Ringing B} }
    p { true }
    a { {DialingA=false; IdleB=false} }
}

30. IdleADialingB CalledACallingB {
    i { {Dial B A} }
    o { {AudibleRinging B,Ringing A} }
    p { true }
    a { { IdleA=false; DialingB=false } }
}

31. BusyABusyB IdleABusyB {
    i { {On-hook A} }
    o { { } }
    p { true }
    a { { IdleA=true } }
}

32. CalledACallingB IdleAIdleB {
    i { {On-hook B} }
    o { { } }
    p { true }
    a { { term=true;IdleA=true;IdleB=true} }
}

33. ConnectedCalledAConnectedCallerB HoldCalledAHoldCallerB
    i { {On-hook A} }
    o { { } }
    p { true }
    a { { } }
}

34. BusyAIdleB BusyADialingB {
    i { {Off-hook B} }
    o { {DialTone B} }
    p { true }
    a { { IdleB=false; DialingB=true } }
}

35. LimboAIdleB DialingAIdleB {
    i { {Timeout A} }
    o { {DialTone A} }
    p { true }
    a { { DialingA = true; } }
}

36. IdleALimboB DialingALimboB {
    i { {Off-hook A} }
    o { {DialTone A} }
    p { true }
    a { { IdleA=false;DialingA=true} }
}

37. LimboAIdleB LimboADialingB {
    i { {Off-hook B} }
    o { {DialTone B} }
    p { true }
    a { { IdleB=false;DialingB=true} }
}

```

FIG. 6



```

43. DialingALimboB IdleALimboB {
    i { {On-hook A} }
    o { {} }
    p { true }
    a { {IdleA=true;DialingA=false} }
}

44. BusyALimboB IdleALimboB {
    i { {On-hook A} }
    o { {} }
    p { true }
    a { {IdleB=true} }
}

45. BusyALimboB IdleALimboB {
    i { {On-hook A} }
    o { {} }
    p { true }
    a { {term=true;IdleA=true} }
}

46. LimboADialingB LimboAIdleB {
    i { {On-hook B} }
    o { {} }
    p { true }
    a { {DialingB=false;IdleB=true} }
}

47. LimboABusyB LimboAIdleB {
    i { {On-hook B} }
    o { {} }
    p { true }
    a { {IdleB=true} }
}

48. CalledACallingB ConnectedCalledAConnectedCallerB {
    i { {Off-hook A} }
    o { {} }
    p { true }
    a { {} }
}

49. DialingALimboB BusyALimboB {
    i { {Dial A B} }
    o { {LineBusyTone A} }
    p { true }
    a { {DialingA=false} }
}

```

R/G. 7

```

50. DialingADialingB IdleADialingB {
    i { { On-hook A } }
    o { { } }
    p { true }
    a { { IdleA=true; DialingA=false } }
}

51. LimboADialingB LimboABusyB {
    i { { Dial B A } }
    o { { LineBusyTone B } }
    p { true }
    a { { DialingB=false } }
}

52. CallingACalledB LimboAIdleB {
    i { { Timeout AB } }
    o { { } }
    p { true }
    a { { IdleB=true; } }
}

53. BusyAIdleB DialingAIdleB {
    i { { Timeout A } }
    o { { DialTone A } }
    p { true }
    a { { DialingA=true } }
}

54. DialingADialingB DialingAIdleB {
    i { { On-hook B } }
    o { { } }
    p { true }
    a { { DialingB=false; IdleB=true } }
}

55. BusyADialingB DialingADialingB {
    i { { Timeout A } }
    o { { DialTone A } }
    p { true }
    a { { DialingA=true } }
}

56. ConnectedCalledACalledCallerB LimboAIdleB {
    i { { On-hook B } }
    o { { } }
    p { true }
    a { { IdleB=true } }
}

57. BusyALimboB DialingALimboB {
    i { { Timeout A } }
    o { { DialTone A } }
    p { true }
    a { { DialingA=true } }
}

58. HoldCallerAHoldCalledB ConnectedCallerACalledB {
    i { { Off-hook B } }
    o { { } }
    p { true }
    a { { } }
}

59. BusyABusyB DialingABusyB {
    i { { Timeout A } }
    o { { DialTone A } }
    p { true }
    a { { DialingA=true } }
}

60. CallingACalledB IdleAIdleB {
    i { { Onhook A } }
    o { { } }
    p { true }
    a { { term=true; IdleA=true; IdleB=true } }
}

61. LimboADialingB DialingADialingB {
    i { { Timeout A } }
    o { { DialTone A } }
    p { true }
    a { { DialingA=true } }
}

```

FIG. 8

```

62, LimboABusyB DialingABusyB {
    i { {Timeout A} }
    o { {DialTone A} }
    p { true }
    a { DialingA=true }
}

63, IdleABusyB IdleADialingB {
    i { {Timeout B} }
    o { {DialTone B} }
    p { true }
    a { DialingB=true }
}

64, DialingABusyB DialingADialingB {
    i { {Timeout B} }
    o { {DialTone B} }
    p { true }
    a { DialingB=true }
}

65, DialingALimboB DialingADialingB {
    i { {Timeout B} }
    o { {DialTone B} }
    p { true }
    a { DialingB=true }
}

66, BusyABusyB BusyADialingB {
    i { {Timeout B} }
    o { {DialTone B} }
    p { true }
    a { DialingB=true }
}

67, BusyALimboB BusyADialingB {
    i { {Timeout B} }
    o { {DialTone B} }
    p { true }
    a { DialingB=true }
}

```

F/6. 9

# 1 Generate all acyclic paths

1a. Collapse strongly connected components into a single node

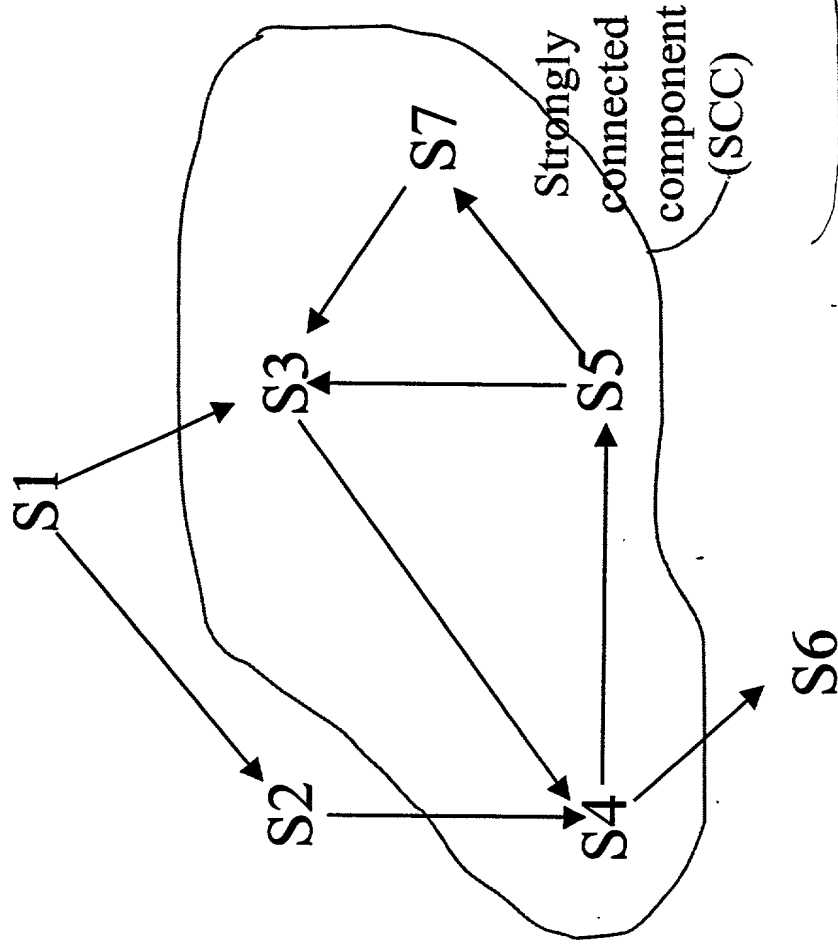
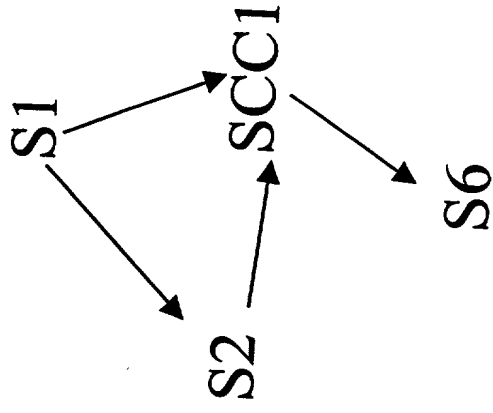


Fig. 10

# 1 Generate all acyclic paths

1b Determine all paths in DAG:

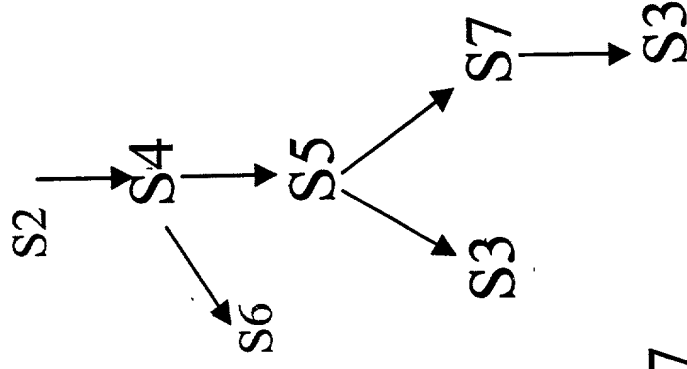
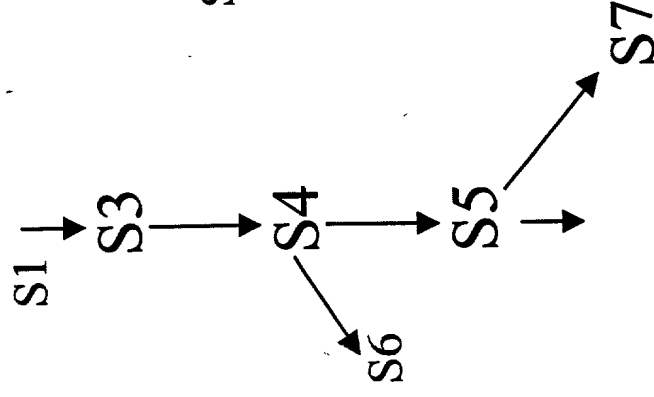
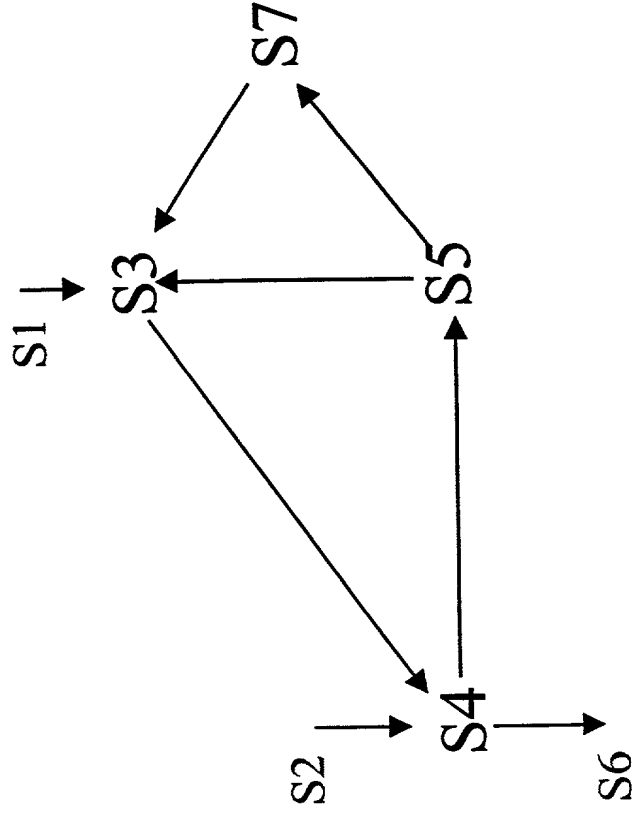


S1, S2, SCC1, S6  
S1, SCC1, S6

*F/G, 11*

# 1 Generate all acyclic paths

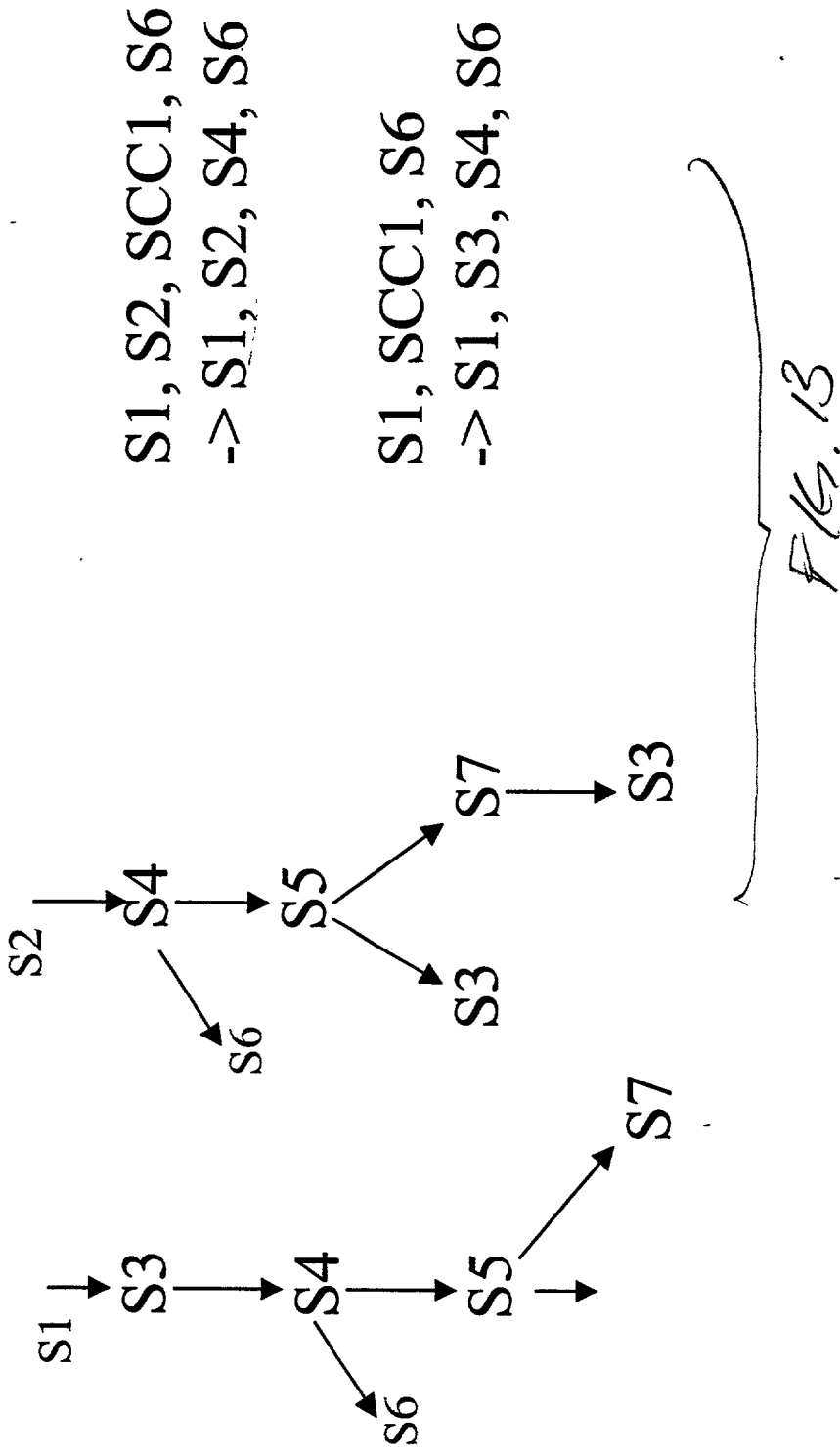
## 1c Build next-transition tree for SCC:



Feb. 12

# 1 Generate all acyclic paths

1d Expand each SCC with all paths through next-transition tree

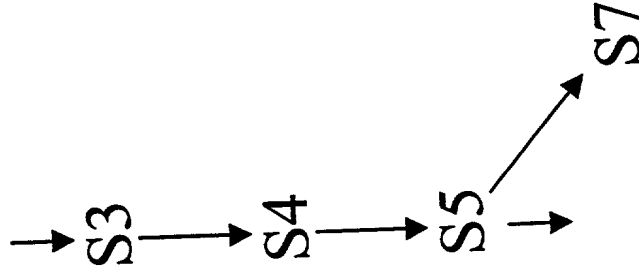


S1, S2, SCC1, S6  
 -> S1, S2, S4, S6

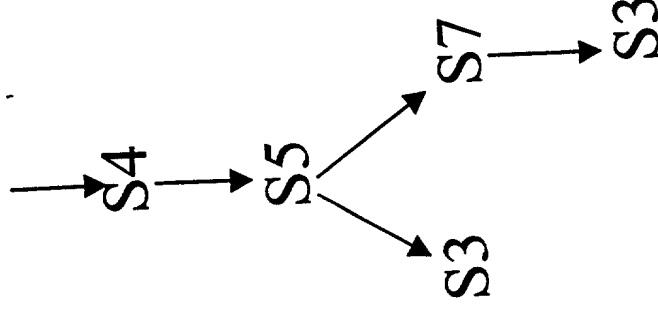
S1, SCC1, S6  
 -> S1, S3, S4, S6

## 2 Generate all simple cycles

Re-use next-transition tree



S3, S4, S5, S3  
S3, S4, S5, S7, S3



S4, S5, S3, S4  
S4, S5, S7, S3, S4

FIG. 14



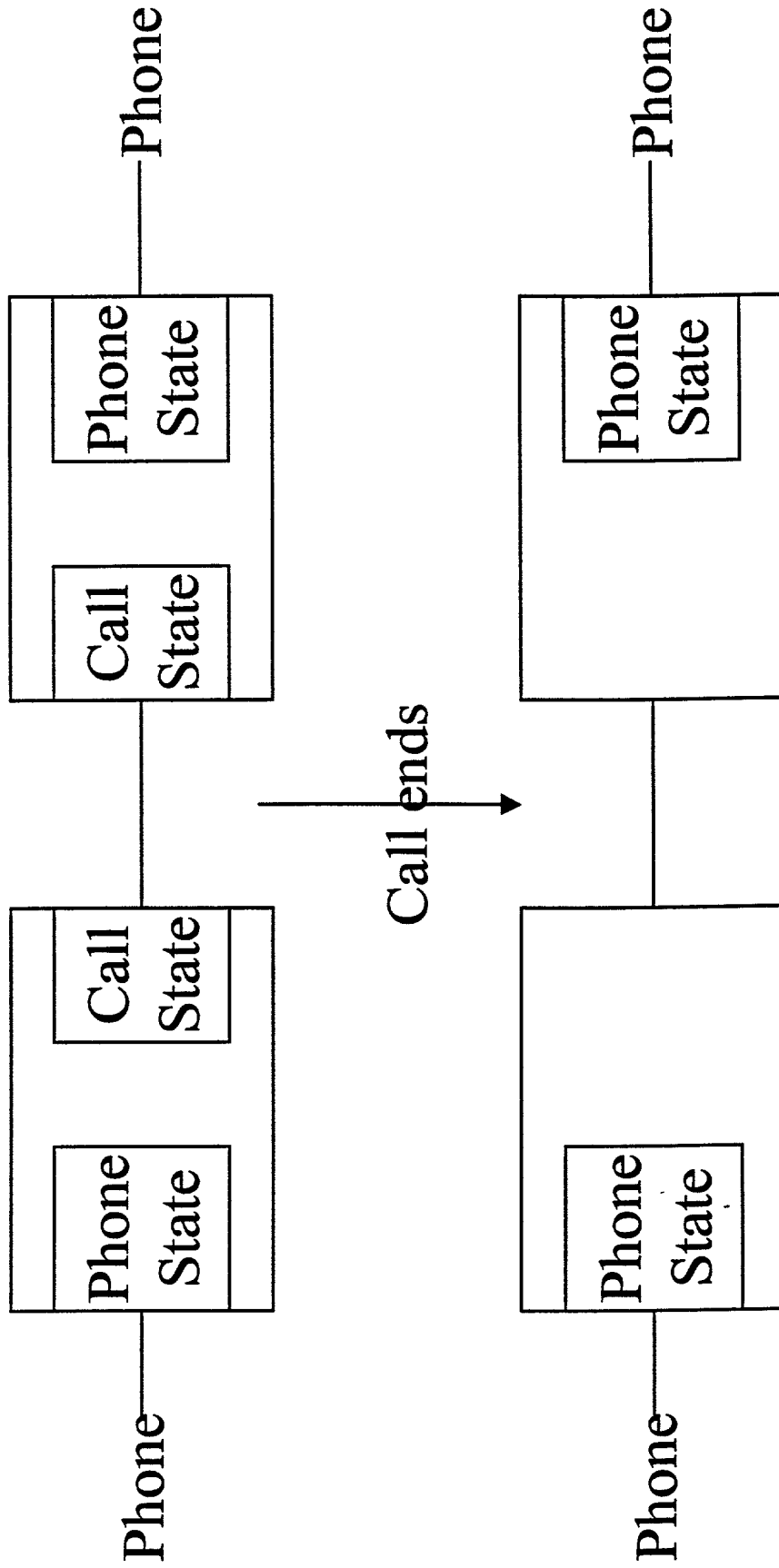
### 3 Combine the paths and the cycles

$$\begin{array}{l} S1, S2, S4, S6 \\ S1, S3, S4, S6 \end{array} + \begin{array}{l} S3, S4, S5, S3 \\ S3, S4, S5, S7, S3 \end{array}$$

$$= \begin{array}{l} S1, S2, S4, S5, S3, S4, S6 \\ S1, S2, S4, S5, S7, S3, S4, S6 \\ S1, S3, S4, S5, S3, S4, S6 \\ S1, S3, S4, S5, S7, S3, S4, S6 \end{array}$$

Fig. 15

# Criteria for adequate coverage



PKS 16

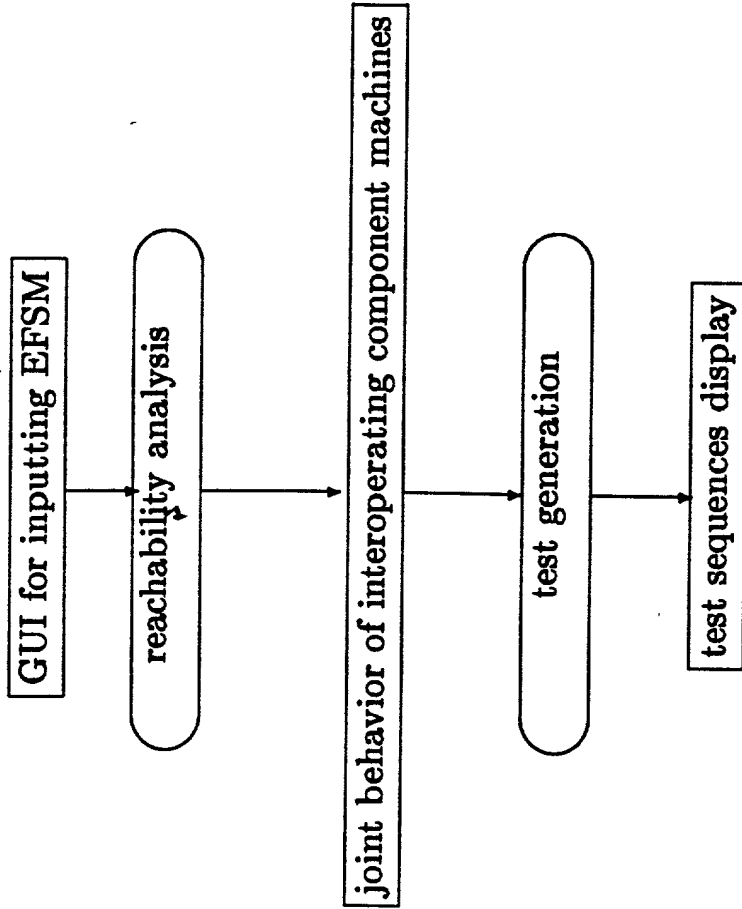


Fig. 17

IN THE UNITED STATES  
PATENT AND TRADEMARK OFFICE

Declaration and Power of Attorney

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name.

I believe I am an original, first and joint inventor of the subject matter which is claimed and for which a patent is sought on the invention entitled **GENERATION OF TEST SUITES FOR INTEROPERABILITY OF REACTIVE COMMUNICATION SYSTEMS** the specification of which is attached hereto.

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by an amendment, if any, specifically referred to in this oath or declaration.

I acknowledge the duty to disclose all information known to me which is material to patentability as defined in Title 37, Code of Federal Regulations, 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, 119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

None

I hereby claim the benefit under Title 35, United States Code, 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, 112, I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

None

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

I hereby appoint the following attorney(s) with full power of substitution and revocation, to prosecute said application, to make alterations and amendments therein, to receive the patent, and to transact all business in the Patent and Trademark Office connected therewith:

|                        |                    |
|------------------------|--------------------|
| Lester H. Birnbaum     | (Reg. No. 25830)   |
| Richard J. Botos       | (Reg. No. 32016)   |
| Jeffery J. Brosemer    | (Reg. No. 36096)   |
| Kenneth M. Brown       | (Reg. No. 37590)   |
| Donald P. Dinella      | (Reg. No. 39961)   |
| Guy Eriksen            | (Reg. No. P-41736) |
| Martin I. Finston      | (Reg. No. 31613)   |
| James H. Fox           | (Reg. No. 29379)   |
| William S. Francos     | (Reg. No. 38456)   |
| Barry H. Freedman      | (Reg. No. 26166)   |
| Julio A. Garceran      | (Reg. No. 37138)   |
| Mony R. Ghose          | (Reg. No. 38159)   |
| Jimmy Goo              | (Reg. No. 36528)   |
| Anthony Grillo         | (Reg. No. 36535)   |
| Stephen M. Gurey       | (Reg. No. 27336)   |
| John M. Harman         | (Reg. No. 38173)   |
| Donald E. Hayes, Jr.   | (Reg. No. 33245)   |
| John W. Hayes          | (Reg. No. 33900)   |
| Michael B. Johannesen  | (Reg. No. 35557)   |
| Mark A. Kurisko        | (Reg. No. 38944)   |
| Irena Lager            | (Reg. No. 39260)   |
| Christopher N. Malvone | (Reg. No. 34866)   |
| Scott W. McLellan      | (Reg. No. 30776)   |
| Martin G. Meder        | (Reg. No. 34674)   |
| Geraldine Monteleone   | (Reg. No. 40097)   |
| John C. Moran          | (Reg. No. 30782)   |
| Michael A. Morra       | (Reg. No. 28975)   |
| Gregory J. Murgia      | (Reg. No. 41209)   |
| Claude R. Narcisse     | (Reg. No. 38979)   |
| Joseph J. Opalach      | (Reg. No. 36229)   |
| Neil R. Ormos          | (Reg. No. 35309)   |
| Eugen E. Pacher        | (Reg. No. 29964)   |
| Jack R. Penrod         | (Reg. No. 31864)   |
| Daniel J. Piotrowski   | (Reg. No. P-42079) |
| Gregory C. Ranieri     | (Reg. No. 29695)   |
| Scott J. Rittman       | (Reg. No. 39010)   |
| Eugene J. Rosenthal    | (Reg. No. 36658)   |

OFFICE OF THE SECRETARY

[illegible]

Inventor's signature Hao ~~Purbing~~ Date 01/28/2000

Date 01/28/2000

**Post Office Address:** 265 Spruce Mill Lane  
Scotch Plains, NJ 07076

Inventor's  
signature\_

David Lee

Date \_\_\_\_\_

1/28/2000

Post Office Address: 13 Rambling Brook Lane  
Warren, NJ 07059

Inventor's  
signature\_\_\_\_\_

**Post Office Address:** 163 South 9th Avenue  
Manville, NJ 08835



**ATTACHMENT A**

Attorney Name: Leo Zucker

Reg. No.: 27,608

Telephone calls should be made to Leo Zucker at:

Phone No.: (914) 761-7799

Fax No.: (914) 761-8767

All written communications are to be addressed to:

Law office of Leo Zucker  
50 Main Street, Suite 480  
White Plains, New York 10606-1975